UILU-ENG 71-2206

AD 720643

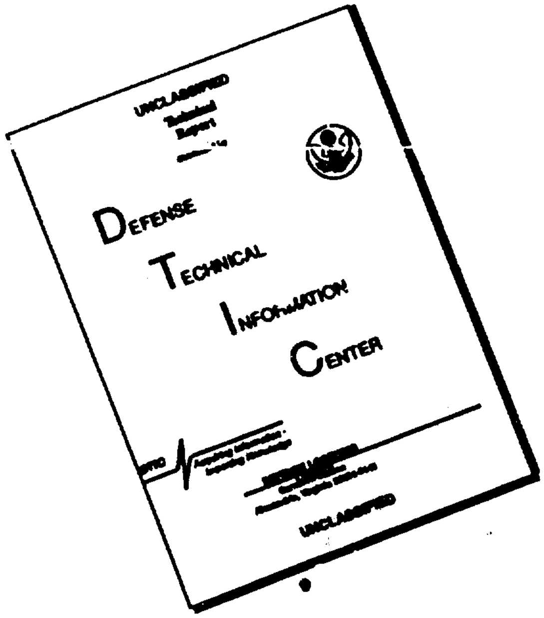
THE IMPLEMENTATION OF AN ALGORITHM FOR THE RETRIEVAL OF PATENT APPLICATIONS

by

Donald Glenn Wagoner

This work was supported by the Joint Services Electronics Program (U. S. Army, U. S. Navy and U. S. Air Force) under Contract DAAB-07-67-C-0199.

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

ACKNOWLEDGEMENT

The author would like to express his gratitude to Professor R. T. Chien for his help and criticisms in the preparation of this thesis, and especially to Ken Biss and Fred Stahl for their suggestions in the writing of this report. Thanks also to Pulavarthi Satyanarayana for his help in debugging the program used in this paper. Also, thanks is extended to Mrs. Sherry Kallembach for the typing of this manuscript.

TABLE OF CONTENTS

I. INTRODUCTION

Due to the technological boom of the last decade, the patent office has found itself in a difficult position. As of 1966, there were three million patents on record This number is increasing at the rate of sixty thousand patents a year [1] With such a large volume of new ideas, it is nearly impossible for a patent examiner to search prior art in order to judge the uniqueness of a new patent. The patent office has therefore been forced to use an automated search routine.

Although automated retrieval systems have greatly reduced the burden for the patent office, they have not solved all of the problems. That is, these systems do not offer the user a natural method of querying the machine.[1] Also, these systems retrieve nonrelevant patents.

In order that a retrieval system be natural, the system must be capable of following a search routine similar to that which the examiner would have done manually In a manual search, the examiner reads the specifications, drawings and statements of claims. From this the examiner chooses those concepts best describing the patent After several prior patents have been selected, the diagrams are then checked for similarities between the patent in question and prior art From this, a judgment value is made of the novelty of the patent in question

With the mechanized search routines now in use, the examiner is only capable of choosing keyword functional concepts Thus the diagrams of patents are not available for usage by this system. With the retrieval method proposed in this report, diagram information will be used in addition to keyword

retrieval. This will add greater flexibility to the system, and in addition this system will be in a more natural form for the user.

Since many keyword retrieval systems have been used in patent retrieval, this report will deal only with the implementation of a retrieval model for circuit diagram information. This system was implemented by using a high level form of pattern recognition. That is, all the loops of a circuit diagram are used as the data base.

In order to gain insight into the effectiveness and details of the proposed recognition model, one must first gain a fundamental understanding of how patents are created. This groundwork will be discussed in Part II. With the necessary groundwork, a study of other retrieval systems and their effectiveness in retrieving relevant documents will be made in Part III Next, a detailed study will be made of the proposed model in Part IV Finally, with an understanding of both the proposed model and other retrieval systems, a comparison of systems will be made in Part V.

The Appendix contains a program of the model algorithm and examples of the program's uses.

The patent retrieval system in this report uses LISP1.5[2,3] as the program language.

II. THE COMPOSITION OF PATENT DOCUMENTS

Patents are legal documants issued by the federal government for new inventions. The purpose of the document is to protect the patentee's invention against plagiarism.

In order to accomplish this goal, patents are subdivided into three fundamental parts.[1] The first part deals with the specifications which consist of general background information, description of the invention and examples of its uses. Since the first part is written with regard to the legal aspects of the document, this portion of the patent will not be considered in the proposed retrieval model.

The second part consists of detailed diagrams of the invention, if any, and relevant test parameters. In the case of patents dealing with circuits, the section contains a complete circuit diagram along with the test parameters. Finally, and most importantly, the third part contains a statement of claims declared by the patentee. "The novelty and patentability are judged by the claims when a patent is granted, questions of infringement are judged by the courts on the basis of the claims."[4]

A patentee may duplicate claims of prior patents as long as not all of the claims are duplicated. That is, the patent in question must contain new ideas.

III. EXISTING PATENT RETRIEVAL SYSTEMS

Several systems have been implemented in the area of patent retrieval. The first to be implemented was the Peek-A-Boo system.[5] This system was created by first placing all patents onto a film matrix. Next, subject cards of keywords are created by punching holes, corresponding to particular patents, into the card. When a subject card is placed over the film matrix, only those patents containing that particular keyword are seen through the key punch holes.

Any number of keywords may be used as the search question in this Peek-A-Boo system. For example, if a user needs information relating to transistor control circuits, the two subject cards, transistors and control, will be selected for the search. When these cards are placed over the film matrix, only those patents relating to transistor control will be visible through the subject cards.

This retrieval system does have the capability of handling functional concepts, but the system also has several problems associated with it. The first problem is the inability to handle diagram information. That is, since patents are also made up of drawings and diagrams, this particular information is lost by the system. The second problem is the possibility of different connotations of keywords by indexer and user. This error will lead to the retrieval of nonrelevant material and the exclusion of some relevant patents.

In order to reduce the problem of keyword interpretation, thesauruses are used.[6] These thesauruses improve the retrieval capabilities of the system, although relevant patents are still missed. In addition, there is a need for continuous updating of the look-up dictionary, which adds to the complexity of the system.

A further improvement in the efficient retrieval of relevant patents is achieved by use of the Smart system.[7] The Smart system uses keyword retrieval and in addition applies complex correlation functional measurements to document and key word. By the application of these functional measurements, ranked documents in order of decreasing relevancy are retrieved from the system That is, a list of documents and associated probability factors are given as output. Also, the Smart system has a routine which will evaluate the efficiency of the thesaurus.

Although the Smart system is effective in retrieving relevant material, the system is very complex to implement. Also this system does not have the capability of handling circuit diagram information. This reduces the flexibility of the system and makes the system less "natural" for usage by the patent examiner.

In order that the complex job of implementing may be reduced, the IBM system uses the technique of total text retrieval.[8,9] This technique may be broken down into three major parts The first part consists of entering the total text of each patent into the computer. Next, the computer indicates the number of times each word occurs. Finally, a technical person selects appropriate keywords from the list given by the computer

Even though the IBM system reduces the complexity of implementing a system, a technical person is still required to update a look-up dictionary Also, there is the added difficulty of decreased efficiency in the retrieval of relevant patents as compared with more complex schemes.

IV. A MODEL FOR DIAGRAMATIC PATENT RETRIEVAL

Simulation programs were first looked at as a possible model for the pattern recognition portion of the retrieval system, but these programs were inflexible.[16,17,18,19] That is, the output is totally dependent on each element of the circuit, and thus this technique does not allow for inclusion. Next, a technique of node retrieval was used as a model.[13,14,15,20,21] Although this technique has only been used for chemical diagrams, an attempt was made to adapt this technique to circuit diagrams. Unfortunately, this algorithm breaks down, as can be seen in figure 1. That is, circuits A and B are identical electrically, but with the use of the nodal algorithm both circuits would be shows as different. The loop concept on the other hand,

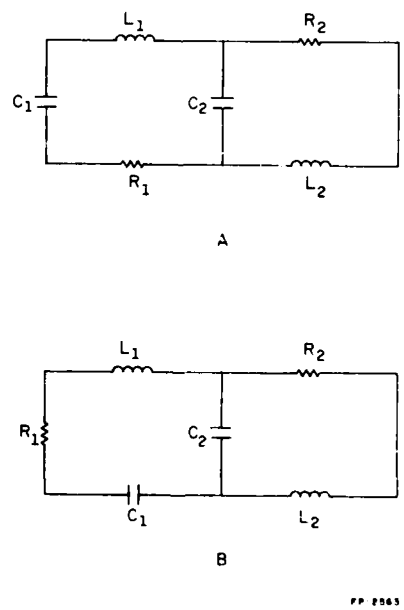


Figure 1

would show both circuits as being the same. The comparison of the algorithm applied to figure 1 is shown in table 1.

ALGORITHM			
NODAL METHOD		LOOP METHOD	
A	B	A	B
(L1 C1)	(L1 C1)	(L1 C1 R1 C2)	(L1 C1 R1 C2)
(R1 C1)	(R1 C1)	(L1 C1 R1 L2 R2)	(L1 C1 R1 L2 R2)
(L1 R2 C2)	(L1 R2 C2)	(R2 C2 L2)	(R2 C2 L2)
(R1 C2 L2)	(C1 C2 L2)		
(R2 L2)	(R2 L2)		

Table 1

Finally, a list of all possible loops of a circuit was used as the retrieving scheme.[10,11,12] Retrieval is then accomplished by matching the loops of the circuit in the search request with those loops already in the data base.

The input to the model is a list of circuit nodes, each node consisting of a list of elements connected to it, along with the number of that particular node. For instance, the circuit in figure 2 would show node one

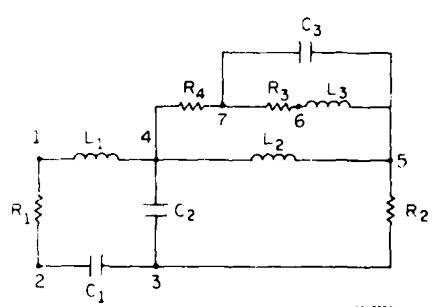


Figure 2

as the list (R1 L1 1). Node two would be indicated by the list (R1 C1 2). The total input data of the circuit would be the list of nodes

((R1 L1 1)(R1 C1 2)(C1C2 R2 3)(L1 C2 L2 R4 4)
(L2 R2 L3 C3 5)(R3 L3 6)(R4 R3 C3 7)).

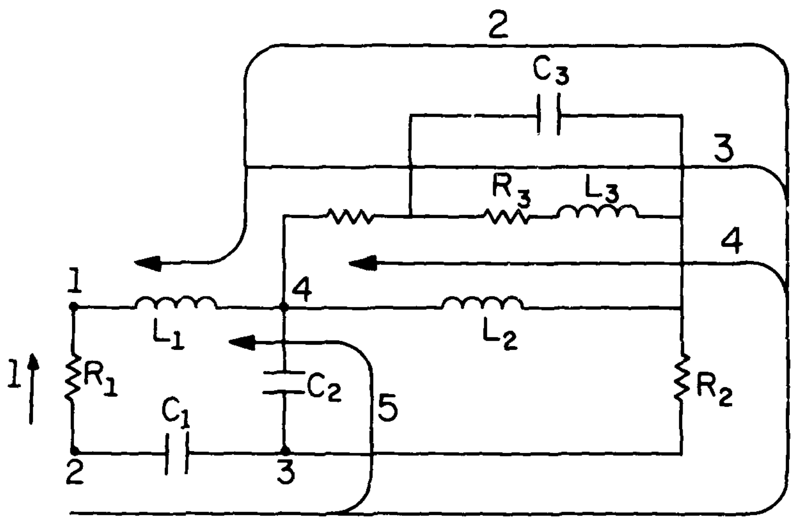
The node and element numbers shown in figure 2 are totally arbitrary. That is, one may give a circuit element or node any number designation. Although the numbers used are arbitrary, the circuit elements must follow the code shown in Table 2. In addition, when using transistor circuits, add the additional node (B_ E_ O_) for each transistor.

ELEMENT	CODE
RESISTOR	R__
INDUCTOR	L__
CAPACITOR	C__
CRYSTAL	X__
TRANSISTOR BASE	B__
TRANSISTOR EMITTER	E__
TRANSISTOR COLLECTOR	O__

Table 2

With the input data now in the computer, the program searches for all possible paths between the first two nodes of the circuit. For the example given in figure 2, the paths are as shown in figure 3.

The Search routine acts like a "mouse through a maze". That is, the computer starts at node two of figure 3 and searches for all possible paths that lead to node one. When the computer reaches node three, it remembers all possible branches out of three. Next, the computer successively tries



FP-2556

Figure 3

each branch to see if it will lead to node one. If it leads to one, then the path is given as output, otherwise, the path is discarded. In the manner described above, all possible paths are checked and only the paths reaching node one will be given as output.

The final answer of the Search routine would be the list of possible paths between node two and one. In the example given, the answer would be

((R1)(C1 R2 C3 R4 L1)
(C1 R2 L3 R3 R4 L1)(C1 R2 L2 L1)(C1 C2 L1))

where (C1 C2 L1) represents path five, etc.

The answer from the Search routine is now applied to the Compare routine. Compare gives all the possible combinations of paths taken two at a time as output. Thus, the Compare function contains a list of all possible

loops of the circuit. For example, if path one and five of figure 3 are combined, the loop (L1 R1 C1 C2) will be created. This result is shown in figure 4.

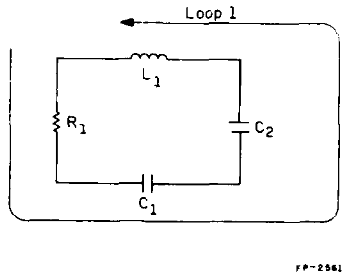
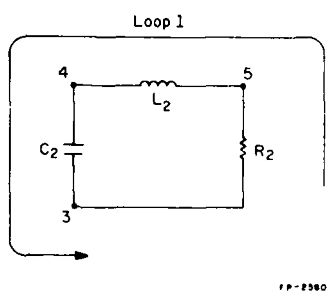


Figure 4

The Compare routine in practice applies the 'exclusive or' to all path combinations. For example, if path four and five of figure 3 are combined via the 'exclusive or' function, the answer will be as shown in figure 5.



PATH 3 $\oplus$ PATH 4 = LOOP 2

(C1 R2 L2 4) $\oplus$ (C1 C2 L1) = (R2 L2 C2)

Figure 5

Next, if all possible path combinations of figure 2 are applied to the 'exclusive or' function, the answer to the Compare routine will then be

((R2 L2 C2)(R2 L3 R3 F4 C2)(L3 R3 F4 L2)(R2 C3 R4 C2)(C3 R4 L2)
(C3 L3 R3)(R1 C1 C2 L1)(R1 C1 R2 L2 L1)(R1 C1 R2 R3 R4 L1))
(R1 C1 R2 C3 R4 L1))

Finally, the answer to the Compare routine is applied to the (Data Base) routine. Data Base will take each loop of the Compare routine and code it into a number. As an example, if R represents the number 100, C represents 10 and L represents 1, the loop (R1 C1 C2 L1) will be coded as the number 121.

$$(1)100 + (2)10 + (1)1 = 121$$

That is, R, L and C are substituted for their corresponding numbers and added together. The coding numbers that are used for the model program is shown in Table 3.

ELEMENT	CODE NUMBER
RESISTOR	1 0 0 0 0 0 0 0
CAPACITOR	1 0 0 0 0 0
INDUCTOR	1 0 0 0 0
TRANSISTOR BASE	1 0 0 0
TRANSISTOR EMITTER	1 0 0
TRANSISTOR COLLECTOR	1 0
CRYSTAL	1

Table 3

With the above coding scheme, each loop can have a maximum of 99 resistors, 99 capacitors, 9 inductors, 9 transistors and 9 crystals. These numbers are quite sufficient for any practical patent circuit. If the coding scheme is now applied to the example in Figure 2, the answer will be

(10110000 30110000 20020000
20200000 10110000 10110000
10210000 20120000 40120000
30210000)

The discussion to this point has dealt with the indexing of circuit diagrams. Now a detailed explanation of the retrieval portion of the program will be examined.

The retrieval portion of the model program first applies the above indexing algorithm to the search circuit. The loops of the search circuit are then compared with those in the data base. If all loops of the search circuit are members of any patent in the data base, those patent numbers are retrieved from memory.

The retrieval algorithm is similar to that used by IBM.[9] However, it differs in that retrieval is done by applying Boolean relations to diagrams rather than keywords. In this particular system, the intersection is taken between the search circuit loops and the data base loops. If this result returns as an answer the original list of search circuit loops, the particular patent is retrieved from memory. That is, if the conditions of Figure 6 exist, the data base patent will be retrieved.

$$A \subseteq B$$

A = The list of search circuit loops

B = The list of loops from a database patent

Figure 6

The number of retrieved patents will be inversely proportional to the complexity of the search circuit. This situation is analogous to key word retrieval systems. That is, the higher the number of keywords used, the smaller the number of retrieved patents. For the proposed model, the higher the number of elements in the search circuit, the smaller the number of retrieved patents. To use the total retrieval system, the user would select keywords best describing the circuit function and a portion of the patent circuit diagram as the search circuit. Patents retrieved by both the circuit diagram and keywords would have the highest probability of being relevant. Depending upon the number of retrieved patents, the keywords and/or search circuit would be modified such that only a few patents are retrieved. As with any other system, the final decision of the uniqueness of the patent in question rests with the user. These decisions are usually judgment values and thus cannot be implemented on the computer.

V. COMPARISON WITH OTHER RETRIEVAL SYSTEMS

All the previous retrieval systems described in this paper use keyword retrieval. This technique has the principle problem of conceptual misinterpretation between indexer and user. Unlike these systems, the retrieval of circuit diagrams does not contain the problem of conceptual ambiguity. For example, the element resistor has an unambiguous meaning from circuit to circuit. Unfortunately, diagram retrieval is too specific and thus does not allow for function concepts. A combination of both keyword and circuit diagram retrieval will be able to handle both functional concepts and specific diagram information. Thus this combined system will give the user a greater degree of flexibility than with either system separately. Also, this combined system is in a form that is similar to the manner in which the user would retrieve patents manually. That is, the user first selects keywords best describing the patent, and then the circuit diagrams of the retrieved patents are compared with the patent in question for possible similarities. This system will be able to retrieve more relevant patents. That is, when both keyword and diagram retrieval is used, the intersection of the retrieved patents by both systems will have a greater possibility of relevancy than with either system separately.

VI. SUMMARY

The system described in this paper has many advantages over previously used retrieval systems. The first advantage is that this system can handle diagram information in addition to keyword concepts. Secondly, there is the advantage of greater flexibility of usage by the user. Thirdly, there is the advantage of the capability of retrieving more relevant documents. That is, since the system in this paper uses all the information from the patent (i.e., both the diagram and written words), relevant patent information is less likely to be lost by this system and thus a greater likelihood of retrieving relevant patents exists.

Although this system is an improvement in the retrieval of patents, some relevant patents will still be missed by this system. Also, this system will not allow for hunches or browsing by the examiner. That is, since the above system is completely literal, the user does not have the capability of "stretching" a keyword category such that it could include new concepts of new and different inventions. The final system for patent retrieval must contain these capabilities.

REFERENCES

1. Bryais, H. L. and Cornog, J. R., Search Methods Used with Transistor Patent Applications, IEEE Spectrum, February, 1966, pp. 116-121., New York.

2. McCarthy, J., Abrahams, P. W., Edwards, D. T., Hart, T. P., Levin, M. I., Lisp 1.5 Programmer's Manual, Cambridge, MIT Press, 1962.

3. Weissman, Clark, Lisp 1.5 Primer, Dickerson Publishing Company, Inc. Belmont, California, 1968.

4. U.S. Department of Commerce Patent Office, General Information Concerning Patents, Washington, D.C., July, 1968.

5. Callow, J. H., and Perrett, J. R., Peek-A-Boo Cards by Computer, Information Retrieval ICIREPAT fifth annual meeting, Washington, D.C., 1966, pp. 133-144.

6. Wall, Eugene, Information Retrieval Thesauri Engineers Joint New York Council, Nov. 1962.

7. Salton, Gerald, Automatic Information Organization and Retrieval, McGraw-Hill Book Company, 1968, pp. 423-429.

8. Kaufman, Samuel, The IBM Information Retrieval Center - (ITIRC) System Techniques and Applications, Proceedings of 21st National Association for Computing Machinery, Washington, D.C., 1966, pp. 505-512.

9. Magnino, J. J., Jr., Prime: A Computer Searching System Using Normal Text, Information Retrieval ICIREPAT fifth annual Meeting, Washington, D. C., 1966, pp. 201-219.

10. Kahn, A. B., Topological Sorting of Large Networks, ACM Communications Vol. 5, 1962, pp. 558-562.

11. Welch, J. T., Jr., A Mechanical Analysis of the Cyclic Structure of Undirected Linear Graphs, ACM Journal, Vol. 13, 1966, pp. 205-210.

12. Unger, S. H., GIT-A heuristic Program for Testing Pairs of Directed Line Graphs for Isomorphism, ACM Communications Vol. 7, 1964, pp. 26-34.

13. Van Dam, A. and Evans, D., A Compact Data Structure for Storing, Retrieving and Manipulating Drawings, AFIPS Conf. Proc. Vol. 30, 1967, pp. 601-611.

14. Wexelblat, R. L. and Freedman, H. A., The Multilang On-Line Programming System, AFIPS Conf. Proc. Vol. 30, 1967, pp. 559-571.

15. Sutherland, I. E., Sketchpad A Man-Machine Graphical Communication System, AFIPS Conf. Proc. Vol. 23, 1963, pp. 329-347.

16. Balaban, P., HYPAC-A Hybrid-Computer Circuit Simulation Program, AFIPS Conf. Proc. Vol. 35, 1969, pp. 771-781.

17. Morris, S. M. and Schiesser, W. E., Salem-A Programming System for the Simulation of Systems Described by Partial Differential Equations, AFIPS Conf. Proc. Vol. 33, part 1., 1968, pp. 353-359.

18. Springer, T. E., TAF-A Steady State, Frequency Response, and Time Response Simulation Program, AFIPS Conf. Proc. Vol. 33, part 1., 1968, pp. 359-371.

19. Sedore, S. R., Sceptre: A Programming for Automatic Network Analysis, IBM Journal of Research and Development Vol. 11, No. 6, Nov., 1967, pp. 627-638.

20. Tauber, S. J., Digital Handling of Chemical Structures and Associated Information, ACM 20th National Proceedings, 1965.

21. Leontein, S., A Report on a Study Based on the Tables for U.S. Steroid Punched Cards, Information Retrieval ICIREPAT Fifth Annual Meeting, 1966, pp. 325-334.

APPENDIX A

PROGRAM OF MODEL

```
DEFINE  ((      ( DELETE1    ( LAMBDA ( A L )
( COND   (       ( NULL L)  NIL )
(   ( EQUAL  A  ( CAR L )    )  ( DELETE1  A ( CDR L )   ))
(  T   ( CONS   ( CAR L )    ( DELETE1 A ( CDR L ) )))    )))
( INTERSECT   ( LAMBDA  ( X Y )  (  COND (    ( NULL X )  NIL )
(   ( MEMBER   ( CAR X ) Y )  ( CONS  ( CAR X )
 ( INTERSECT   ( CDR X )   Y )))
(  T  ( INTERSECT   ( CDR X ) Y )   ))))
 ( UNIONS   ( LAMBDA   ( X Y )   ( COND   (   ( NULL X )  Y)
(    ( MEMBER   ( CAR X )   Y )
( UNIONS   ( CDR   X )   Y ))
(  T  (  CONS   ( CAR X )   ( UNIONS   ( CDR X )   Y  )))    )))
 ( DELETE   ( LAMBDA   (X Y )   ( MAPCAR  Y    ( FUNCTION
( LAMBDA   (A)   ( DELETE1 X A )   )))   ))
( MATCH    ( LAMBDA   ( A X )   ( COND   (   ( NULL X )  NIL )
(   ( NULL   ( CAR X ))  NIL )
(    ( MEMBER  A   ( CAR X )    )   ( DELETE1 A   ( CAR X )   ))
 (  T   ( MATCH    A   ( CDR X )))     )))
 ( ELIMINATE   ( LAMBDA   ( A Z )   ( COND   (    ( NULL Z )  NIL )
(   ( EQUAL   A   ( CAR Z )   )
( ELIMINATE   A   ( CDR Z )))
(    T (   CONS   ( CAR Z )   ( ELIMINATE   A   ( CDR Z ))    ))    )))
 ( SUBS1   ( LAMBDA   (X)   ( COND   (   ( NULL X )  NIL )
((ATOM   ( CAR X)) ( SUBS1   ( CDR X )))   ( T X )    )))
( SUBS2   ( LAMBDA   (X)   ( COND   (    ( NULL X )  NIL )
( ( NUMBERP ( CADAR X )) X )
(    ( ATOM   ( CAR   X )   )   ( SUBS2   ( CDR X )   ))
( T ( CONS ( CDAR X ) ( CDR X )   ))   )))    ))
  DEFINE     ((
( SUBS3 ( LAMBDA (X) ( COND (( NUMBERP ( CADAR X ))
( CONS ( CAAR X ) ( CDR X )   ))   ( T X )   )))
 ( EXCLUR   ( LAMBDA   ( A B)   ( EXCLUDE
 ( INTERSECT   A B )   ( UNIONS   A B )   ))   )
( COMPARE ( LAMBDA (X) ( PROG ( Y N Z )
  SETQ Y ( CDR X ))
( SETQ N NIL )
( SETQ Z X )
TAG0
( COND (( NULL Y )  ( RETURN ( CONS ( CAR Z )  NIL ))))  ( T
        ( SETQ N ( CONS ( EXCLUR ( CAR Z ) ( CAR Y )) N ))))
( COND ( ( NULL ( CDR Y ))   ( GO TAG1 ))
( T ( SETQ Y ( CDR Y )   )) )
( GO TAG0 )
TAG1 ( SETQ Z ( CDR Z )   )
( SETQ Y ( CDR Z )   )
( COND (   ( NULL Y )   ( RETURN N ))
( T ( GO TAG0 )   ))   )))
( LAST   ( LAMBDA   (L)   ( COND   (   ( NULL L ) NIL )
 (   ( NULL   ( CDR L )   )   ( CAR L )   )
 ( T ( LAST   ( CDR L )))    )))
( DELETEB ( LAMBDA ( M Z ) ( COND (( NULL M ) Z )
( T ( LAST ( MAPCAR M ( FUNCTION
( LAMBDA   (A)   ( COND   (   ( ATOM  A )
( SETQ  Z ( DELETE1  NIL   ( DELETE  A Z )))   )
 ( T   ( SETQ  Z   ( DELETE1   NIL   ( DELETEA   A Z ))    ))    ))))) ))))
  ( ELIM   ( LAMBDA   ( X Y )   ( COND (   ( NULL X )  Y )
(   ( ATOM   ( CAR X )   )   ( COND (   ( EQUAL   ( INTERSECT
```

```
( CONS  ( CAR X )  NIL )  Y )  ( CONS  ( CAR X )  NIL ))
( QUOTE A )  )  ( T ( ELIM  ( CDR X )  Y ))  ))
(  ( EQUAL  ( INTERSECT  ( CAR X )  Y )  ( CAR X )  )
  ( QUOTE  A )  )
( T ( ELIM  ( CDR X )  Y ))   )))
 ( ELIMINATEA  ( LAMBDA  ( C D )  ( PROG  (P)
( SETQ  P ( MAPCAR  D ( FUNCTION
( LAMBDA  (X)  ( ELIM C X )  ))) )
 ( SETQ  P (  DELETE1  ( QUOTE A )  P ))
( SETQ  P  ( DELETE1  NIL  P ))
 ( RETURN P ) )))
  ( EXCLUDE  ( LAMBDA  ( X Y )  ( COND  (  ( NULL Y )  NIL )
( (  NUMBERP  ( CAR Y )) Y )
(   ( MEMBER  ( CAR Y )   X )  ( EXCLUDE X  ( CDR Y )  ))
 ( T ( CONS  ( CAR Y )  ( EXCLUDE X ( CDR Y )  )))   )))
 ( DELETEA  ( LAMBDA  ( A B )  ( MAPCAR B ( FUNCTION
( LAMBDA (X) ( COND (( NUMBERP ( CAR ( EXCLUDE A X ))) NIL )
( T ( EXCLUDE A X ))  )  )))  ))
(  SEARCHA  ( LAMBDA  (X)  ( PROG  ( Y Z M C L P )
( SETQ  Z  ( CDDR X )  )
( SETQ  Y  ( CAR X )  )
( COND  (  ( NULL  ( INTERSECT Y  ( CADR X )  ))  ( GO TAGO )  ))
(SETQ  L  ( INTERSECT  Y ( CADR X )   ))
( SETQ  Y  ( EXCLUDE  ( INTERSECT  Y  ( CADR X ))  Y ))
TAGO ( COND (  ( NULL Y ) ( GO TAG100 ))
(( NUMBERP ( CAR Y ) ) ( GO TAG100 ) )
( ( NUMBERP ( CADR Y )) ( GO TAG1 )) ( T ( GO TAG2 ) ))
 TAG1  ( COND  (  ( INTERSECT Y ( CADR X ) ) ( GO TAG10 )  ))
( SETQ  M ( CONS  ( CAR Y ) M ))
  ( SETQ  Z ( ELIMINATE  Y Z )  )
 ( COND  (  ( NULL  ( MATCH  ( CAR Y )  Z ))  ( GO TAG100 )) )
 ( SETQ  Y ( MATCH  ( CAR Y )  Z  ))
 ( SETQ   Z  (  DELETE  ( CAR M )  Z ))
 (  GO TAGO )
TAG2  ( COND  (  ( NULL ( INTERSECT Y ( CADR X )  ))
( GO  TAG3 )  ))
TAG10  ( COND  (  ( NULL ( CDR ( INTERSECT Y ( CADR X )  )))
 ( SETQ  P ( CONS  ( CAR ( INTERSECT Y ( CADR X )  ))
 ( MAPCAR  M  ( FUNCTION  ( LAMBDA  (A)
 ( COND  (   ( ATOM  A )  A )
 ( T ( CAR A ))  ))))    ))
   ( T ( SETQ P  ( MAPCAR ( INTERSECT Y ( CADR X ) )
( FUNCTION  ( LAMBDA (A)
 ( CONS A ( MAPCAR  M ( FUNCTION ( LAMBDA (A)
  ( COND  (   ( ATOM A )  A )  ( T ( CAR A )))   ))) ) )))   )) ))
( SETQ  Z ( DELETEA  ( INTERSECT  Y  ( CADR X ))  Z ))
 ( SETQ Y ( EXCLUDE  ( INTERSECT Y ( CADR X )) Y ))
 ( SETQ  C  ( CONS  P C )  )
 ( GO TAGO )
TAG3  ( SETQ  M ( CONS  Y M ))
  ( SETQ  Z  ( DELETEA  ( CDR Y )  Z ) )
( SETQ Y ( CONS ( CAR Y ) ( LAST Y )  ))
  ( SETQ  Z ( ELIMINATE  Y Z )  )
( COND  (  ( MATCH  ( CAR Y ) Z )
  ( SETQ  Y ( MATCH  ( CAR Y )  Z )  ))
 ( T  ( GO TAG100  ))  )
( SETQ  Z ( DELETE  ( CAAR M )  Z ))
```

```
( GO TAGO  )
TAG100  ( SETQ  Z  ( CDDR X )  )
( SETQ  M  ( SUBS1  M )  )
 ( COND  (  ( NULL M )  ( RETURN  ( CONS L C )  )))
( SETQ  Z ( ELIMINATEA  M Z )  )
 ( SETQ  M  ( SUBS2  M ))
 ( COND  (  ( NULL M )  ( RETURN  ( CONS L C )  )))
 ( SETQ  Z ( DELETEB  ( CDR M )  Z ) )
( SETQ  M  (  SUBS3  M )  )
(  COND  (  ( ATOM  ( CAR M )  )  ( COND  (  ( NULL ( MATCH
( CAR M )  Z ))  ( GO TAG100  ))
( T (  SETQ  Y ( MATCH  ( CAR M )  Z )  )) ))
(  T ( COND  (  ( NULL  ( MATCH  ( CAAR M )  Z  ))
( GO TAG100  ))  ( T ( SETQ  Y ( MATCH  ( CAAR M )  Z )))
)))
 ( COND  (  ( ATOM  ( CAR M )  )  ( SETQ Z ( DELETE
  ( CAR M )  Z ))  )
  ( T ( SETQ  Z ( DELETE  ( CAAR M )  Z ))  ) )
  ( GO TAG(  )  )))  ))
   DEFINE  ((
( FUG ( LAMBDA   (X) ( CAR ( EXPLODE ( CAR X )))  ))
( DBASE ( LAMBDA (A)
( PROG ( N)
 ( SETQ  N 0 )
TAG5 ( COND (( EQ ( FUG A ) ( QUOTE R )) ( SETQ N
( PLUS 10000000 N )))
(( EQ ( FUG A ) ( QUOTE C )) ( SETQ N ( PLUS 100000 N )))
(( EQ ( FUG A ) ( QUOTE L )) ( SETQ N ( PLUS 10000 N )))
(( EQ ( FUG A ) ( QUOTE B )) ( SETQ N ( PLUS 1000 N )))
(( EQ ( FUG A ) ( QUOTE E )) ( SETQ N ( PLUS 100 N )))
 (( EQ ( FUG A ) ( QUOTE O )) ( SETQ N ( PLUS 10 N )))
 (( EQ ( FUG A ) ( QUOTE X )) ( SETQ N ( PLUS 1 ))) )
 ( COND  (  ( NULL ( CDR A )      )  ( RETURN N ))
 ( T ( SETQ  A ( CDR A )  )))
  ( GO TAG5 )   )))
 ( DATABASE  ( LAMBDA (X)  ( MAPCAR X ( FUNCTION
  ( LAMBDA (A)  ( DBASE A )  )  ))))
( DATABASEB ( LAMBDA (X) ( PAIR I2  ( MAPCAR X ( FUNCTION
( LAMBDA (A) ( DATABASE ( COMPARE A ))) )))  ))
( DATABASEA ( LAMBDA (X) ( PAIR I   ( MAPCAR X ( FUNCTION
( LAMBDA (A) ( DATABASE ( COMPARE ( SEARCHA A ))))) )))  )
 ( DATABASEC ( LAMBDA (X) ( DATABASE ( COMPARE ( SEARCHA X )))  ))
( DATABASED ( LAMBDA (X) ( DATABASE ( COMPARE X ))  ))
 ( RETRIEV ( LAMBDA ( J2 ) ( PROG (N)
 ( SETQ N NIL )
( CSETQ J1 J7 )
 ( CSET ( QUOTE J3 ) (  DATABASEC J2 ))
 TAG11
 ( COND (( NULL J1 ) ( RETURN ( PRINT N )  ))
 ( ( EQUAL ( INTERSECT  J3 ( CAR J1 )) J3 )
 ( SETQ N ( CONS ( CAAR J1 ) N )) )
  ( T T )  )
 ( CSETQ J1  ( CDR J1 )  )
 ( GO TAG11 )  )))
 ( RETRIEVE  ( LAMBDA  ( )  ( PROG ( J2 )
TAG53 ( SETQ J2 ( READ ) )
 ( COND (( NULL J2 ) ( RETURN NIL )))
```

APPENDIX B

USER'S MANUAL

1. Indexing of Patents

   (a) CSET list of data base patent to I.

   (b) Make a list of nodes for each patent.

   (c) Make a list of patents.

   (d) Apply data base A function to c

2. Retrieval of Patents

   (a) Enter Retrieve ( )

   (b) Make a list of nodes for each search circuit.

   (c) Place each search circuit on a separate card.

   (d) Last card must read NIL.

APPENDIX C

PROGRAM EXAMPLES

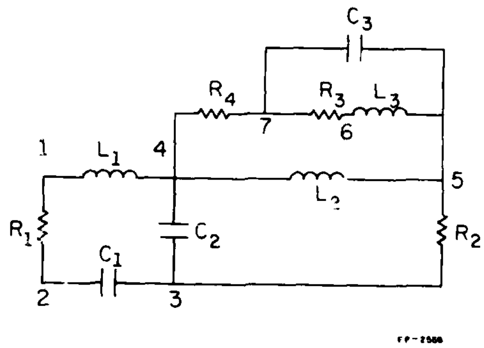
Appendix C contain the following information in the order listed below:

(1) Program example used in the text

(2) Example Patent Data Base

(3) Search Circuits and Retrieved Patents from Data Base

Below is the computer output for the example given in the text.



Patent 100

*** ARGUMENTS OF SEARCHA

(((R1 L1 1) (R1 C1 2)(C1 C2 R2 3)(L1 C2 L2 R4 4)
(L2 R2 L3 C3 5) (R3 L3 6) (R4 R3 C3 7)))

*** VALUE OF SEARCHA

((R1) (C1 R2 C3 R4 L1) (C1 R2 L3 R3 R4 L1) (C1 R2 L2 L1) (C1 C2 L1))

Circuit paths of circuit

*** ARGUMENTS OF COMPARE

(((R1) (C1 R2 C3 R4 L1) (C1 R2 L3 R3 R4 L1) (C1 R2 L2 L1) (C1 C2 L1)))

*** VALUE OF COMPARE

((R2 L2 C2) (R2 L3 R3 R4 C2) (L3 R3 R4 L2) (R2 C3 R4 C2) (C3 R4 L2)
(C3 L3 R3) (R1 C1 C2 L1) (R1 C1 R2 L2 L1) (R1 C1 R2 L3 R3 R4 L1) (R1 C1
R2 C3 R4 L1))

Circuit loops of circuit

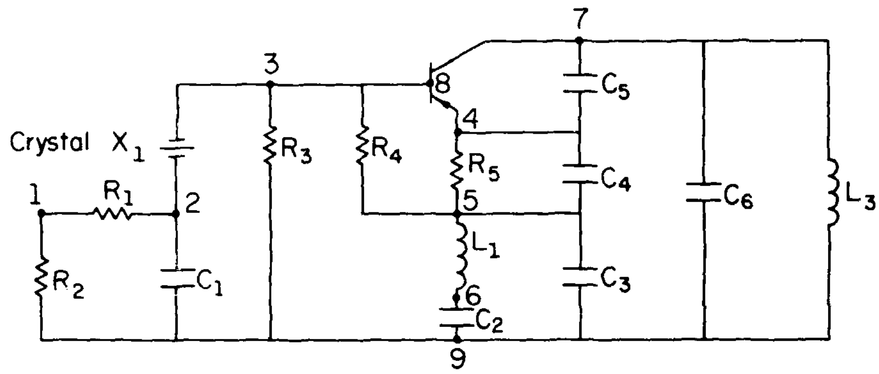
*** ARGUMENTS OF DATABASE

(((R2 L2 C2) (R2 L3 R3 R4 C2) (L3 R3 R4 L2) (R2 C3 R4 C2) (C3 R4 L2)
(C3 L3 R3) (R1 C1 C2 L1) (R1 C1 R2 L2 L1) (R1 C1 R2 L3 R3 R4 L1)
(R1 C1 R2 C3 R4 L1)))

*** VALUE OF DATABASE

(10110000 30110000 20020000 20200000 10110000 10110000 10210000 20120000
40120000 30210000)

Database list



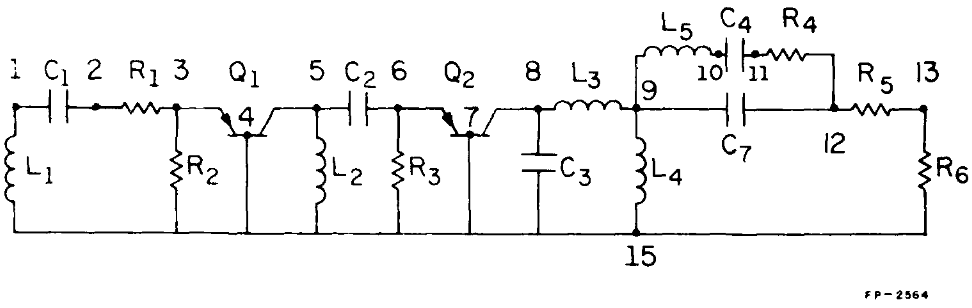
FP-2887

Patent 1000

```
ARGUMENTS FOR EVALQUOTE ...
   DATABASE
   (((R1 R2 1) (R1 C1 X1 2) (X1 R3 R4 B1 3) (R5 C4 C5 E1 4) (R4 R5 L1 C4 C3 5) (L1 C2 6) (C5 C6 L3
   C1 7) (B1 E1 C1 8) (R2 C1 R3 C2 C3 C6 L3 9)))

*** ARGUMENTS OF SEARCHA
   (((R1 R2 1) (R1 C1 X1 2) (X1 R3 R4 B1 3) (R5 C4 C5 E1 4) (R4 R5 L1 C4 C3 5) (L1 C2 6) (C5 C6 L3
   C1 7) (B1 E1 C1 8) (R2 C1 R3 C2 C3 C6 L3 9)))
 COLLECTED     27222   CELLS AND STACK HAS      2978   UNITS LEFT.
 COLLECTED     28533   CELLS AND STACK HAS      2859   UNITS LEFT.
 COLLECTED     28484   CELLS AND STACK HAS      2872   UNITS LEFT.
 COLLECTED     28496   CELLS AND STACK HAS      2912   UNITS LEFT.
 COLLECTED     28494   CELLS AND STACK HAS      2882   UNITS LEFT.
 COLLECTED     28499   CELLS AND STACK HAS      2931   UNITS LEFT.
 COLLECTED     28447   CELLS AND STACK HAS      2946   UNITS LEFT.
 COLLECTED     28472   CELLS AND STACK HAS      2924   UNITS LEFT.
 COLLECTED     28440   CELLS AND STACK HAS      2929   UNITS LEFT.
 COLLECTED     28425   CELLS AND STACK HAS      2911   UNITS LEFT.
 COLLECTED     28403   CELLS AND STACK HAS      2884   UNITS LEFT.
 COLLECTED     28431   CELLS AND STACK HAS      2894   UNITS LEFT.
 COLLECTED     28433   CELLS AND STACK HAS      2931   UNITS LEFT.

*** VALUE OF SEARCHA


   ((R1) (X1 R4 C4 E1 C1 L3 R2) (X1 B1 C1 L3 R2) (X1 B1 E1 C5 L3 R2) (X1 R4 C4 C5 L3 R2) (X1 R4 C4
   E1 C1 C6 R2) (X1 B1 C1 C6 R2) (X1 B1 E1 C5 C6 R2) (X1 R4 C4 C5 C6 R2) (X1 B1 E1 C4 C3 R2) (X1 B1
   C1 C5 C4 C3 R2) (X1 B1 E1 R5 C3 R2) (X1 B1 C1 C5 R5 C3 R2) (X1 R4 C3 R2) (X1 B1 E1 C4 L1 C2 R2)
    (X1 B1 C1 C5 C4 L1 C2 R2) (X1 B1 E1 R5 L1 C2 R2) (X1 B1 C1 C5 R5 L1 C2 R2) (X1 R4 L1 C2 R2) (X1
   C2 R2) (C1 R2))

*** ARGUMENTS OF COMPARE
   (((R1) (X1 R4 C4 E1 C1 L3 R2) (X1 B1 C1 L3 R2) (X1 B1 E1 C5 L3 R2) (X1 R4 C4 C5 L3 R2) (X1 R4 C4
   E1 C1 C6 R2) (X1 B1 C1 C6 R2) (X1 B1 E1 C5 C6 R2) (X1 R4 C4 C5 C6 R2) (X1 B1 E1 C4 C3 R2) (X1 B1
   C1 C5 C4 C3 R2) (X1 B1 E1 R5 C3 R2) (X1 B1 C1 C5 R5 C3 R2) (X1 R4 C3 R2) (X1 B1 E1 C4 L1 C2 R2)
    (X1 B1 C1 C5 C4 L1 C2 R2) (X1 B1 E1 R5 L1 C2 R2) (X1 B1 C1 C5 R5 L1 C2 R2) (X1 R4 L1 C2 R2) (X1
   C2 R2) (C1 R2)))
 COLLECTED     28133   CELLS AND STACK HAS      2962   UNITS LEFT.
 COLLECTED     27604   CELLS AND STACK HAS      2914   UNITS LEFT.
 COLLECTED     27066   CELLS AND STACK HAS      2921   UNITS LEFT.
```

```
- * VALUE OF COMPARE
((X] R3 C1) (X] R4 L1 C2 C1) (R4 L1 C2 R3) [illegible]
[illegible]
F] O] L3 R2))
COLLECTED      26824    CELLS AND STACK HAS       2943   UNITS LEFT.


TIME    45637MS,  VALUE IS ...
([illegible]
[illegible]
10311011 23113111)
--- END OF DATA
```

EXAMPLE PATENT DATABASE

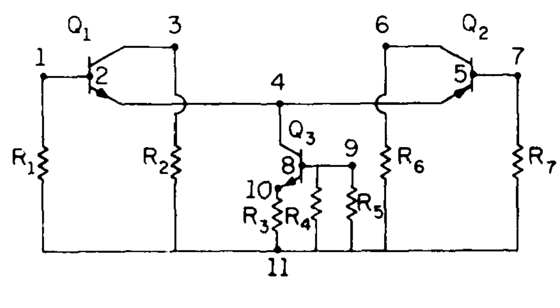


PATENT 2000

ARGUMENTS FOR EVALQUOTE ...

```
SEARCHA
(((L1 C1 1) (C1 R! 2) (R1 R2 E1 3) (B1 E1 01 4) (01 L2 C2 5) (C2 Re E2 6)
(B2 E2 02 7) (02 C3 L3 8) (L3 L4 C7 L5 9) (L5 C4 10) (C4 R4 11) (C7 R4
 R5 12) (R5 R6 E3 13) (E3 B3 03 14) (L1 R2 B1 L2 R3 B2 C3 L4 $6 B4 15)))

TIME   68956MS, VALUE IS ...
 ((C1) (R1 E1 01 C2 E2 02 Le L5 C4 R4 R5 R6 L1) (R1 E1 01 C2 E2 02 L3
 C7 R5 R6 L1) (R1 E1 01 C2 E2 02 L3 L4 L1) (R1 E1 01 C2 E2 02 C3 L1)
 (R1 F1 01 C2 E2 B2 L1) (R1 E1 01 C2 R3 L1) (R1 E1 01 L2 L1) (R1 E1 B1 L!)
 (R1 R2 L1))
```

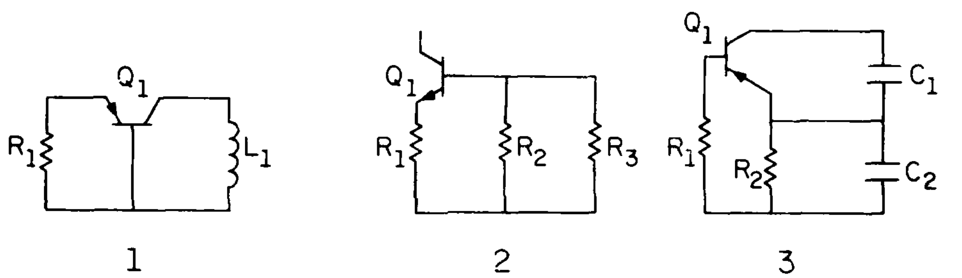
EXAMPLE PATENT DATABASE



PATENT 3000

```
ARGUMENTS FOR EVALQUOTE ...
   SEARCHA
   (((R1 B1 1) (B1 E1 O1 2) (O1 R2 3) (E1 E3 O2 4) (B3 E3 O3 5) (O3 R6 6)
   (B3 R7 7) (B2 E2 B2 R4 R5 9) (E2 R3 10) (R1 R2 R3 R4 R5 R6 R7 11)))
               26615   CELLS AND STACK HAS    2944   UNITS LEFT.
               28526   CELLS AND STACK HAS    2869   UNITS LEFT.
               28503   CELLS AND STACK HAS    2950   UNITS LEFT.

TIME    9002MS,    VALUE IS ...
  ((B1) (E1 E3 B3 R7 R1) (E1 E3 O3 R6 R1) (E1 O2 B2 R5 R1) (E1 O2 B2 R4 R1)
  (E1 O2 E2 R3 R1) (O1 R2 R1))
```

Three SEARCH Circuits and RETRIEVE
PATENTS FROM DATABSE




```
ARGUMENTS FOR EVALQUOTE ...
  RETRIEVE
  NIL
  (2000) 1
  (3000) 2
  (1000) 3

TIME     1680MS,    VALUE IS ...
  NIL
```

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

Coordinated Science Laboratory University of Illinois Urbana, Illinois 61801	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED
	2b. GROUP

THE IMPLEMENTATION OF AN ALGORITHM FOR THE RETRIEVAL OF PATENT APPLICATIONS

(Type of report and inclusive dates)

(First name, middle initial, last name)

Donald Glenn Wagoner

	7a. TOTAL NO OF PAGES	7b. NO OF REFS
March, 1971	30	21

DAAB-07-67-C-0199	9a. ORIGINATOR'S REPORT NUMBER(S) R-503
	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) UILU-ENG 71-2206

This document has been approved for public release and sale; its distribution is unlimited.

	12. SPONSORING MILITARY ACTIVITY Joint Services Electronics Program through U. S. Army Electronics Command, Fort Monmouth, New Jersey, 07703

Due to the technological boom of the last decade, the patent office has found itself in a difficult position. As of 1966, there were three million patents on record. This number is increasing at the rate of sixty thousand patents a year. With such a large volume of new ideas, it is nearly impossible for a patent examiner to search prior art in order to judge the uniqueness of a new patent. The patent office has therefore been forced to use an automated search routine. A retrieval method is proposed in this report.

DD FORM 1473

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Document Retrieval						
Pattern Recognition						
Keyword						